Enterprise Computing

Stephen Gilmore

The University of Edinburgh

About this course

- This is a eighteen-lecture course at the third-year undergraduate level (Level 9).
- There is a written examination in the April/May diet of examinations.
- There is a single coursework exercise spanning the course. This coursework is undertaken as a group exercise.

The coursework project

- The aim of the coursework is to experience a realistic enterprise computing problem: making an existing legacy application accessible over the Web.
- You will work in teams to achieve this, with direction on requirements for the networked application and requirements on the approved technology to be used.
- You will be required to demonstrate your progress so far on the project at various stages.

Course bias

- Enterprise Computing is a green course (suitable for Software Engineering degrees).
- The focus in this course is on practical skills acquisition rather than abstraction, formalisation and generalisation.
- The aim is to treat the dominant relevant technologies in depth rather than to give a more superficial survey of many rivals.
- This course will use the Extensible Markup Language (XML) and the Java 2 Enterprise Edition Software Development Kit.

Course lectures

- The course lectures are Monday and Thursday (from 9:00 to 9:50) in Room 3218 of the James Clerk Maxwell Building.

Course lectures

- The course lectures are Monday and Thursday (from 9:00 to 9:50) in Room 3218 of the James Clerk Maxwell Building.

Course materials

- The course materials are compiled from several sources. Among these are included SUN Microsystems' Java 2 Enterprise Edition tutorial.

The J2EE™ 1.4 Tutorial

Eric Armstrong

Jennifer Ball

Stephanie Bodoff

Debbie Carson

Ian Evans

Maydene Fisher

Dale Green

Kim Haase

Eric Jendrock

The J2EE 1.4 Tutorial copyright notice

Copyright © 2003 Sun Microsystems, Inc., 4150 Network Circle, Santa Clara, California 95054, U.S.A. All rights reserved.

This distribution may include materials developed by third parties.

Sun, Sun Microsystems, the Sun logo, Java, J2EE, JavaServer Pages, Enterprise JavaBeans, Java Naming and Directory Interface, EJB, JSP, J2EE, J2SE and the Java Coffee Cup logo are trademarks or registered trademarks of Sun Microsystems, Inc. in the U.S. and other countries.

Overview

Today, more and more developers want to write distributed transactional applications for the enterprise and leverage the speed, security, and reliability of server-side technology.

Overview

Today, more and more developers want to write distributed transactional applications for the enterprise and leverage the speed, security, and reliability of server-side technology.

In today's fast-moving and demanding world of e-commerce and information technology, enterprise applications have to be designed, built, and produced for less money, with greater speed, and with fewer resources than ever before.

Java 2 Platform, Enterprise Edition (J2EE)

To reduce costs and fast-track application design and development, Java 2 Platform, Enterprise Edition (J2EE) provides a component-based approach to the design, development, assembly, and deployment of enterprise applications.

The J2EE platform offers a multitiered distributed application model, reusable components, a unified security model, flexible transaction control, and Web services support through integrated data interchange on Extensible Markup Language (XML)-based open standards and protocols.

Advantages of J2EE

Not only can you deliver innovative business solutions to market faster than ever, but your platform-independent J2EE component-based solutions are not tied to the products and application programming interfaces (APIs) of any one vendor.

Advantages of J2EE

Not only can you deliver innovative business solutions to market faster than ever, but your platform-independent J2EE component-based solutions are not tied to the products and application programming interfaces (APIs) of any one vendor.

Vendors and customers enjoy the freedom to choose the products and components that best meet their business and technological requirements.

Distributed multitiered applications

The J2EE platform uses a multitiered distributed application model for enterprise applications.

Distributed multitiered applications

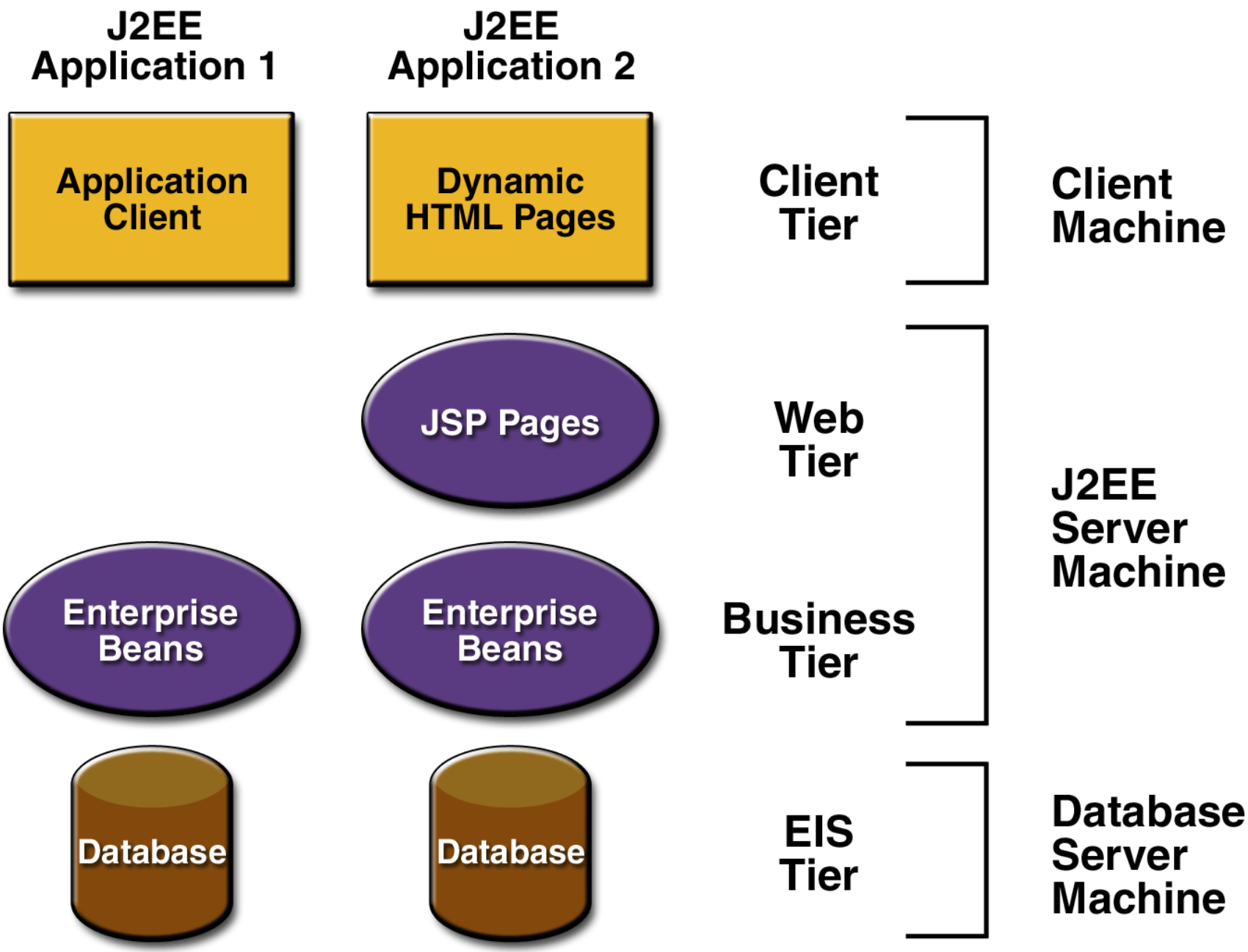
The J2EE platform uses a multitiered distributed application model for enterprise applications.

Application logic is divided into components according to function, and the various application components that make up a J2EE application are installed on different machines depending on the tier in the multitiered J2EE environment to which the application component belongs.

Placement of components on machines

Multitiered J2EE applications can be divided into the tiers described in the following list.

- Client-tier components run on the client machine.
- Web-tier components run on the J2EE server.
- Business-tier components run on the J2EE server.
- Enterprise information system (EIS)-tier software runs on the EIS server.


J2EE Application 1
J2EE Application 2
Application Client
Dynamic HTML Pages
Client Tier
Client Machine
JSP Pages
Web Tier
J2EE Server Machine
Enterprise Beans
Enterprise Beans
Business Tier
Database
Database
EIS Tier
Database Server Machine

Three tiers or four?

Although a J2EE application can consist of the three or four tiers shown, J2EE multitiered applications are generally considered to be three-tiered applications because they are distributed over three different locations: client machines, the J2EE server machine, and the database or legacy machines at the back end.

Three tiers or four?

Although a J2EE application can consist of the three or four tiers shown, J2EE multitiered applications are generally considered to be three-tiered applications because they are distributed over three different locations: client machines, the J2EE server machine, and the database or legacy machines at the back end.

Three-tiered applications that run in this way extend the standard two-tiered client and server model by placing a multithreaded application server between the client application and back-end storage.

J2EE Components

J2EE applications are made up of components.

J2EE Components

J2EE applications are made up of components.

A J2EE component is a self-contained functional software unit that is assembled into a J2EE application with its related classes and files and that communicates with other components.

J2EE Components in the J2EE Specification

The J2EE specification defines the following J2EE components:

- Application clients and applets are components that run on the client.
- Java Servlet and JavaServer Pages™ (JSP™) technology components are Web components that run on the server.
- Enterprise JavaBeans™ (EJB™) components (enterprise beans) are business components that run on the server.

Java programming and J2EE

J2EE components are written in the Java programming language and are compiled in the same way as any program in the language.

Java programming and J2EE

J2EE components are written in the Java programming language and are compiled in the same way as any program in the language.

The difference between J2EE components and "standard" Java classes is that J2EE components are assembled into a J2EE application, verified to be well formed and in compliance with the J2EE specification, and deployed to production, where they are run and managed by the J2EE server.

J2EE Clients

A J2EE client can be a Web client or an application client.

J2EE Clients

A J2EE client can be a Web client or an application client.

Web clients

A Web client consists of two parts: dynamic Web pages containing various types of markup language (HTML, XML, and so on), which are generated by Web components running in the Web tier, and a Web browser, which renders the pages received from the server.

Web clients are thin clients

A Web client is sometimes called a thin client.

Web clients are thin clients

A Web client is sometimes called a thin client.

Thin clients usually do not do things like query databases, execute complex business rules, or connect to legacy applications.

Web clients are thin clients

A Web client is sometimes called a thin client.

Thin clients usually do not do things like query databases, execute complex business rules, or connect to legacy applications.

When you use a thin client, heavyweight operations like these are off-loaded to enterprise beans executing on the J2EE server where they can leverage the security, speed, services, and reliability of J2EE server-side technologies.

Applets

A Web page received from the Web tier can include an embedded applet.

Applets

A Web page received from the Web tier can include an embedded applet.

An applet is a small client application written in the Java programming language that executes in the Java virtual machine installed in the Web browser.

Applets

A Web page received from the Web tier can include an embedded applet.

An applet is a small client application written in the Java programming language that executes in the Java virtual machine installed in the Web browser.

However, client systems will likely need the Java Plug-in and possibly a security policy file in order for the applet to successfully execute in the Web browser.

Web components

Web components are the preferred API for creating a Web client program because no plug-ins or security policy files are needed on the client systems.

Web components

Web components are the preferred API for creating a Web client program because no plug-ins or security policy files are needed on the client systems.

Also, Web components enable cleaner and more modular application design because they provide a way to separate applications programming from Web page design.

Web components

Web components are the preferred API for creating a Web client program because no plug-ins or security policy files are needed on the client systems.

Also, Web components enable cleaner and more modular application design because they provide a way to separate applications programming from Web page design.

Personnel involved in Web page design thus do not need to understand Java programming language syntax to do their jobs.

Application clients

A J2EE application client runs on a client machine and provides a way for users to handle tasks that require a richer user interface than can be provided by a markup language.

Application clients

A J2EE application client runs on a client machine and provides a way for users to handle tasks that require a richer user interface than can be provided by a markup language.

It typically has a graphical user interface (GUI) created using the Swing API.

Application clients

A J2EE application client runs on a client machine and provides a way for users to handle tasks that require a richer user interface than can be provided by a markup language.

It typically has a graphical user interface (GUI) created using the Swing API.

Application clients directly access enterprise beans running in the business tier. However, a J2EE application client can open an HTTP connection to establish communication with a servlet running in the Web tier.

JavaBeans™ Component Architecture

The server and client tiers might also include components based on the JavaBeans component architecture (JavaBeans component) to manage the data flow between an application client or applet and components running on the J2EE server or between server components and a database.

JavaBeans™ Component Architecture

The server and client tiers might also include components based on the JavaBeans component architecture (JavaBeans component) to manage the data flow between an application client or applet and components running on the J2EE server or between server components and a database.

JavaBeans components are not considered J2EE components by the J2EE specification.

JavaBeans components

JavaBeans components have instance variables and `get` and `set` methods for accessing the data in the instance variables.

JavaBeans components

JavaBeans components have instance variables and `get` and `set` methods for accessing the data in the instance variables.

JavaBeans components used in this way are typically simple in design and implementation, but should conform to the naming and design conventions outlined in the JavaBeans component architecture.

J2EE Server Communications

Next, let's consider the various elements that can make up the client tier.

J2EE Server Communications

Next, let's consider the various elements that can make up the client tier.

The client communicates with the business tier running on the J2EE server either directly or, as in the case of a client running in a browser, by going through JSP pages or servlets running in the Web tier.

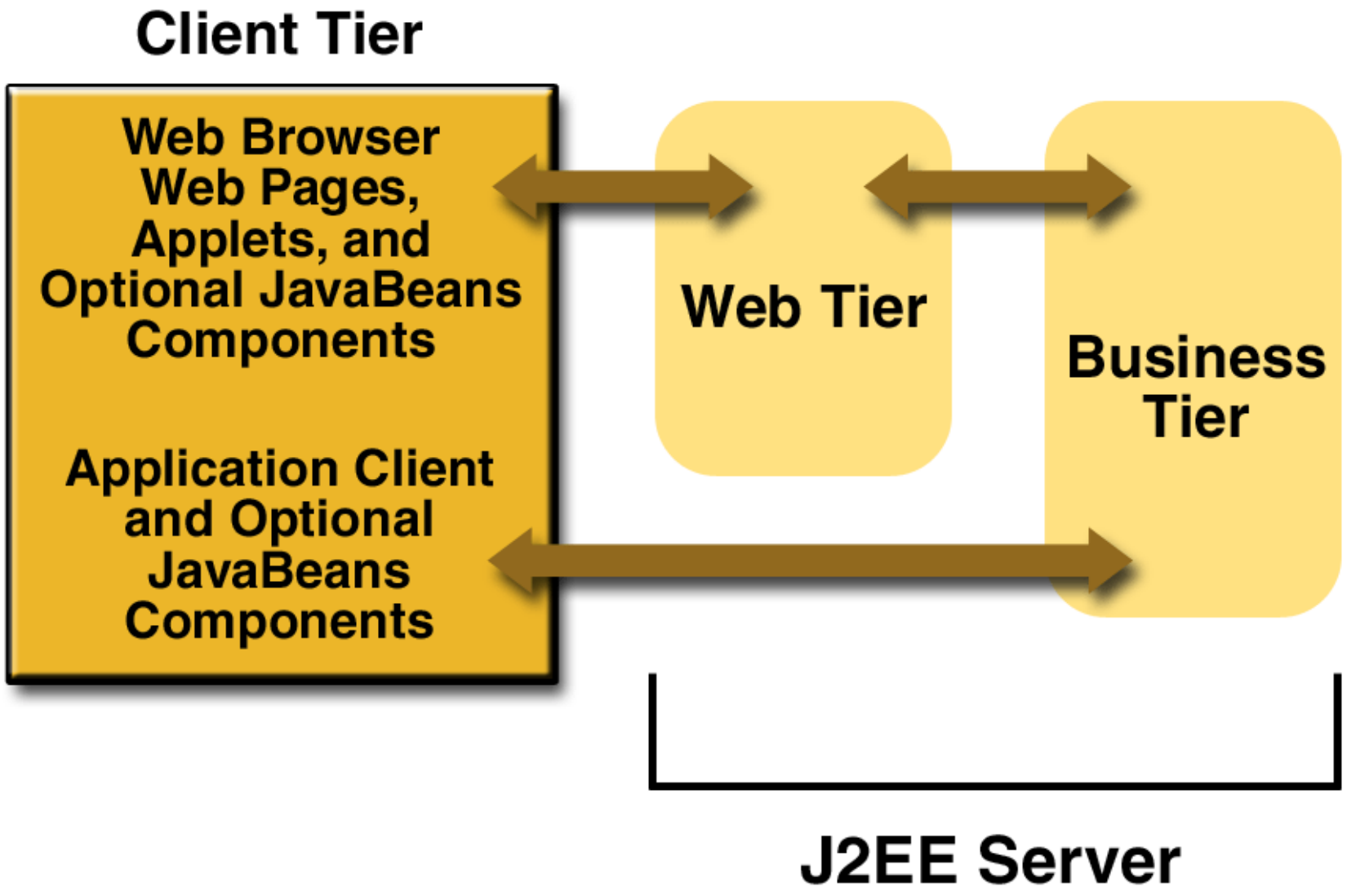
Thin clients versus thick clients

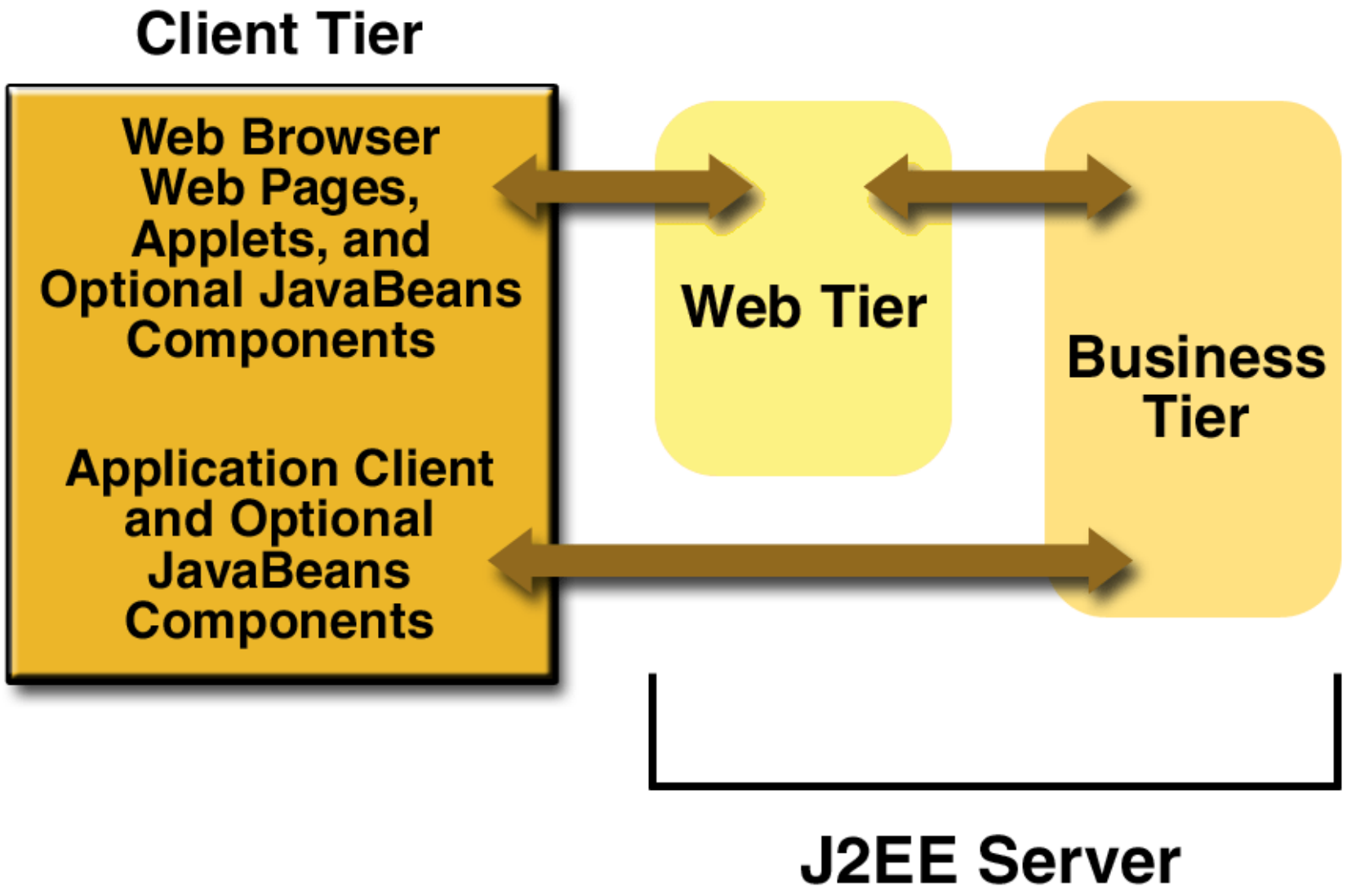
Your J2EE application uses a thin browser-based client or thick application client. To decide which, you should be aware of the trade-offs between keeping functionality on the client and close to the user (thick client) and off-loading as much functionality as possible to the server (thin client).

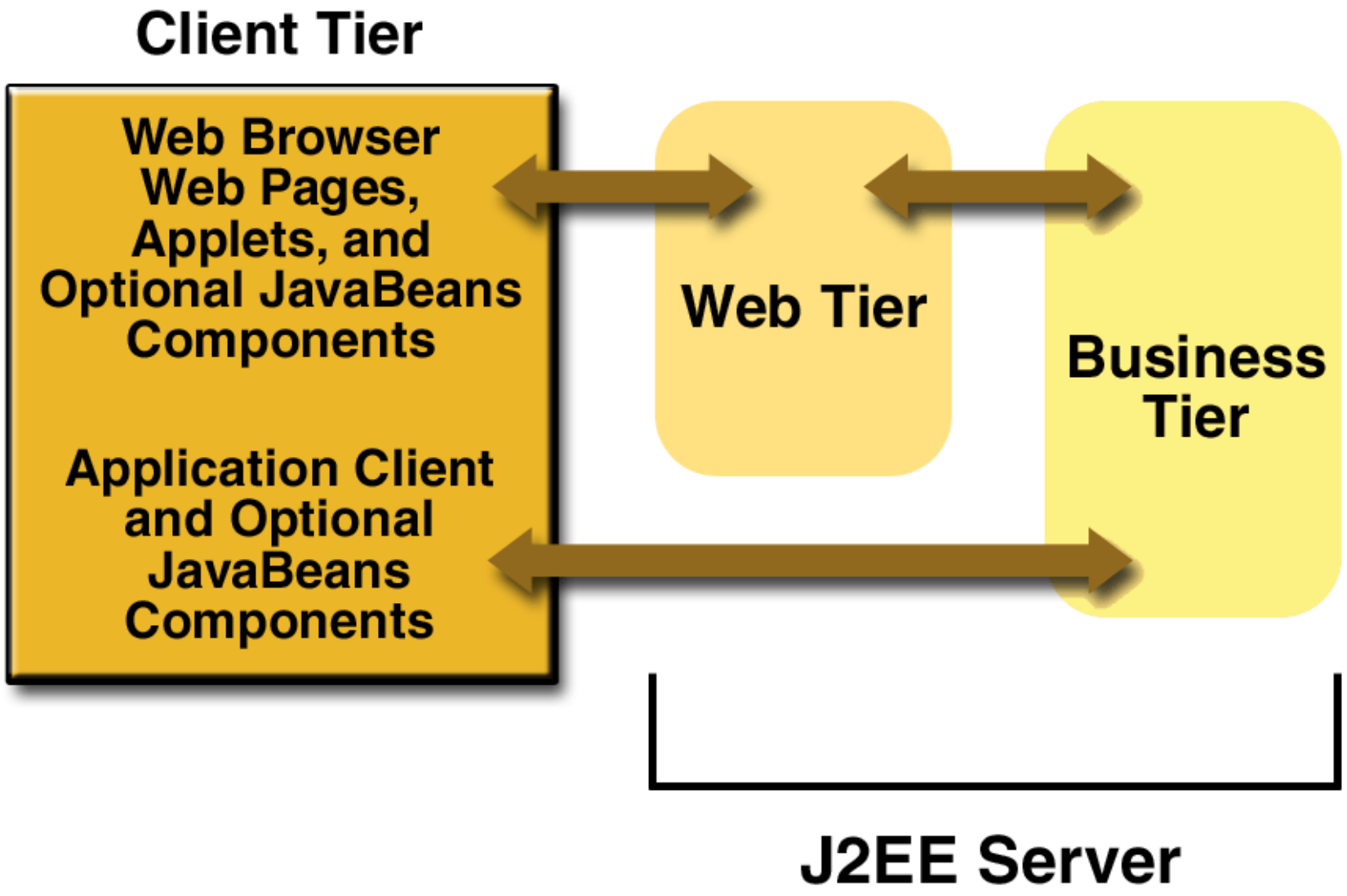
Thin clients versus thick clients

Your J2EE application uses a thin browser-based client or thick application client. To decide which, you should be aware of the trade-offs between keeping functionality on the client and close to the user (thick client) and off-loading as much functionality as possible to the server (thin client).

The more functionality you off-load to the server, the easier it is to distribute, deploy, and manage the application; however, keeping more functionality on the client can make for a better perceived user experience.


Client Tier
Web Browser
Web Pages,
Applets, and
Optional JavaBeans
Components
Application Client
and Optional
JavaBeans
Components
Web Tier
Business
Tier
J2EE Server


Client Tier
Web Browser
Web Pages,
Applets, and
Optional JavaBeans
Components
Application Client
and Optional
JavaBeans
Components
Web Tier
Business
Tier
J2EE Server


Client Tier
Web Browser
Web Pages,
Applets, and
Optional JavaBeans
Components
Application Client
and Optional
JavaBeans
Components
Web Tier
Business
Tier
J2EE Server

Web Components

J2EE Web components can be either servlets or JSP pages.

Web Components

J2EE Web components can be either servlets or JSP pages.

Servlets are Java programming language classes that dynamically process requests and construct responses.

Web Components

J2EE Web components can be either servlets or JSP pages.

Servlets are Java programming language classes that dynamically process requests and construct responses.

JSP pages are text-based documents that execute as servlets but allow a more natural approach to creating static content.

What is not a web component?

Static HTML pages and applets are bundled with Web components during application assembly, but are not considered Web components by the J2EE specification.

What is not a web component?

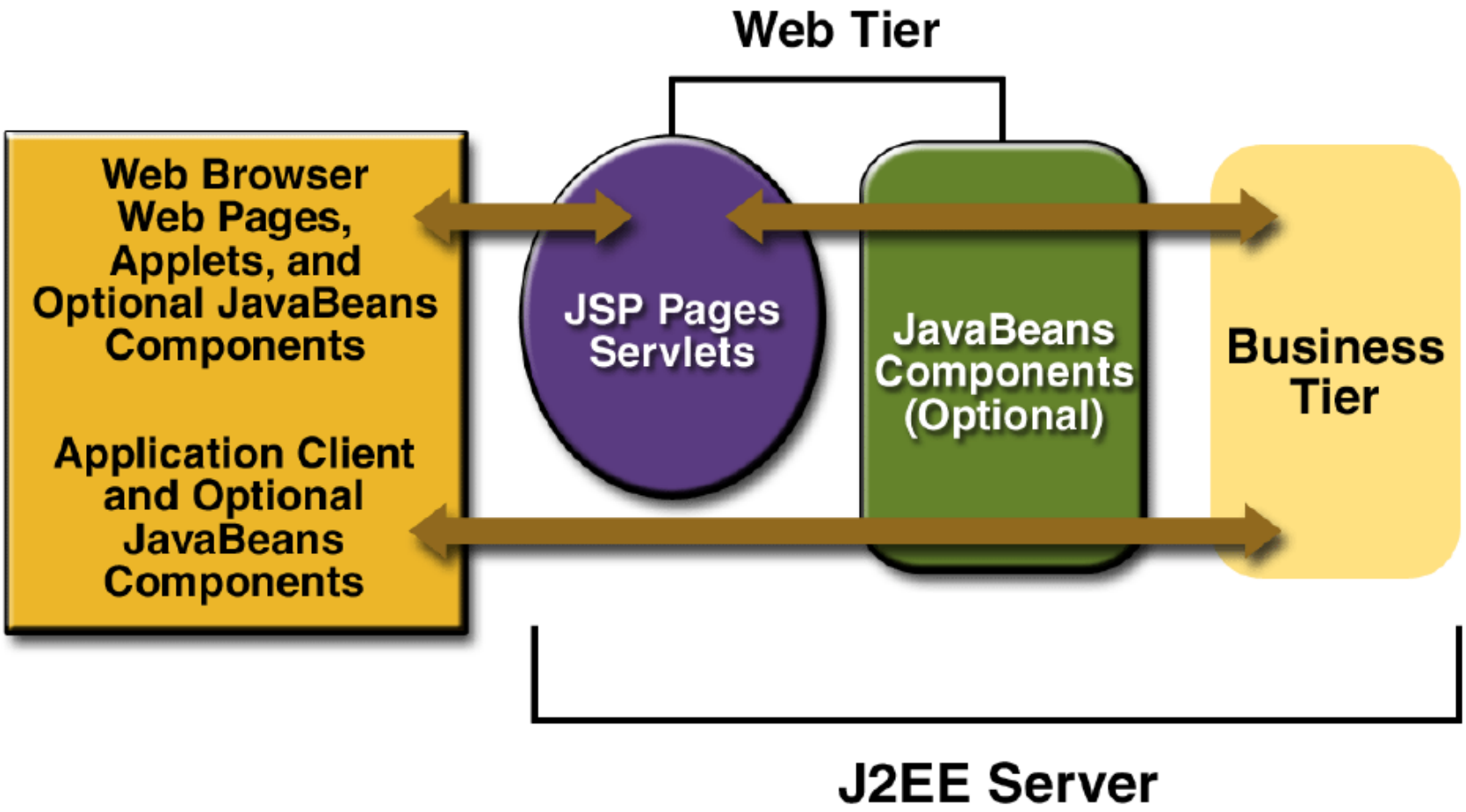
Static HTML pages and applets are bundled with Web components during application assembly, but are not considered Web components by the J2EE specification.

Server-side utility classes can also be bundled with Web components and, like HTML pages, are not considered Web components.

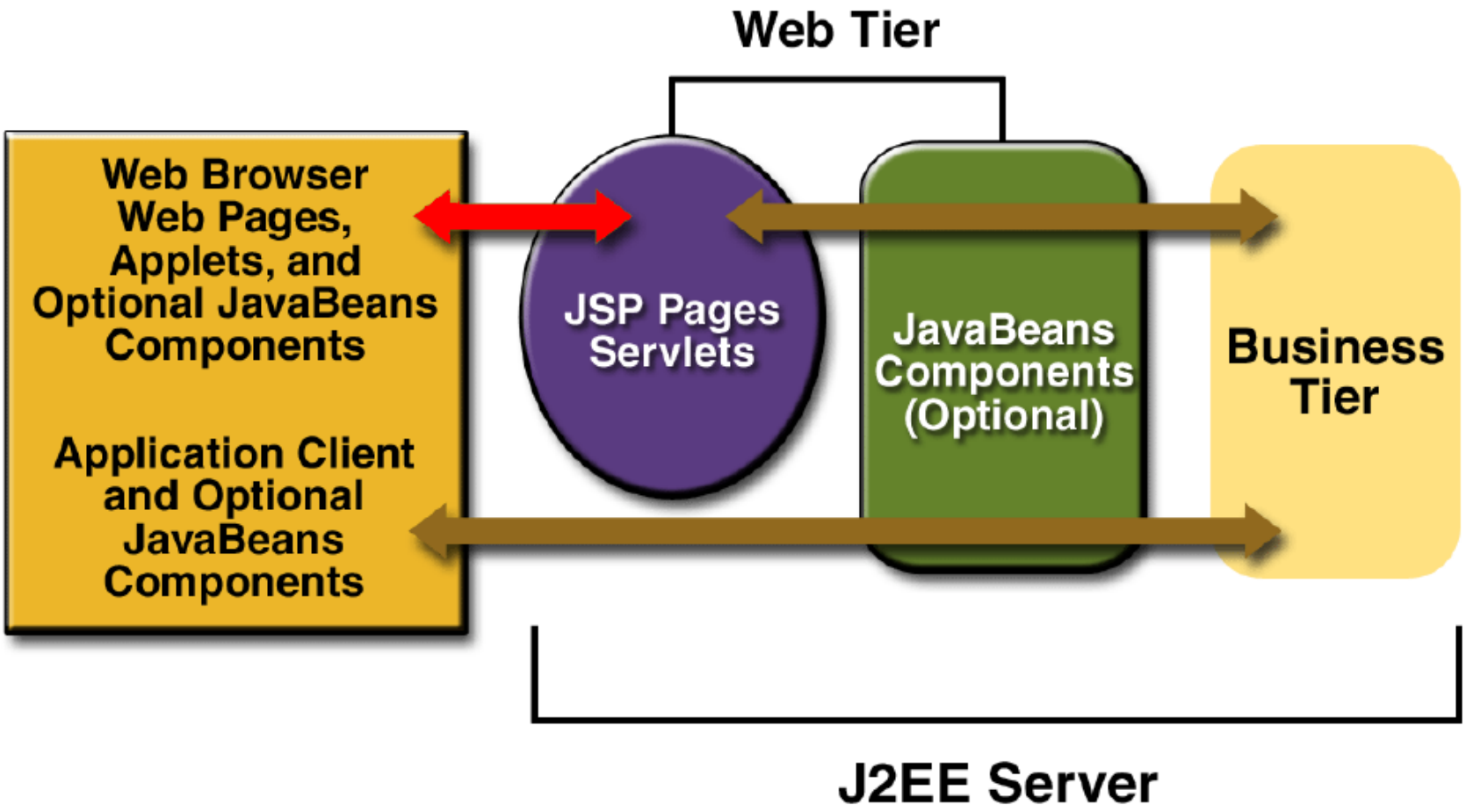
Client tier — Web tier — Business tier

Like the client tier, the Web tier might include a JavaBeans component to manage the user input and send that input to enterprise beans running in the business tier for processing.

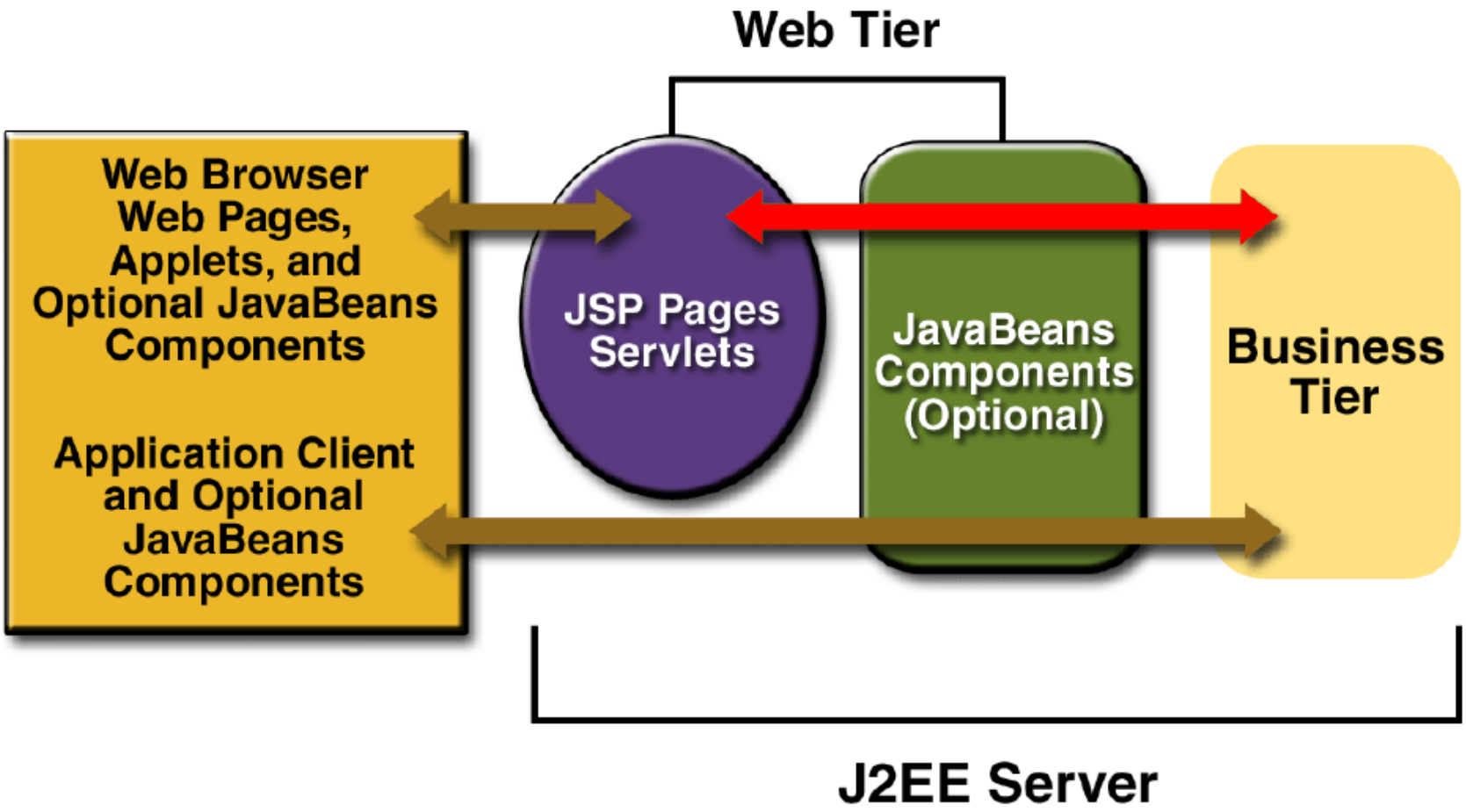
Client tier — Web tier — Business tier



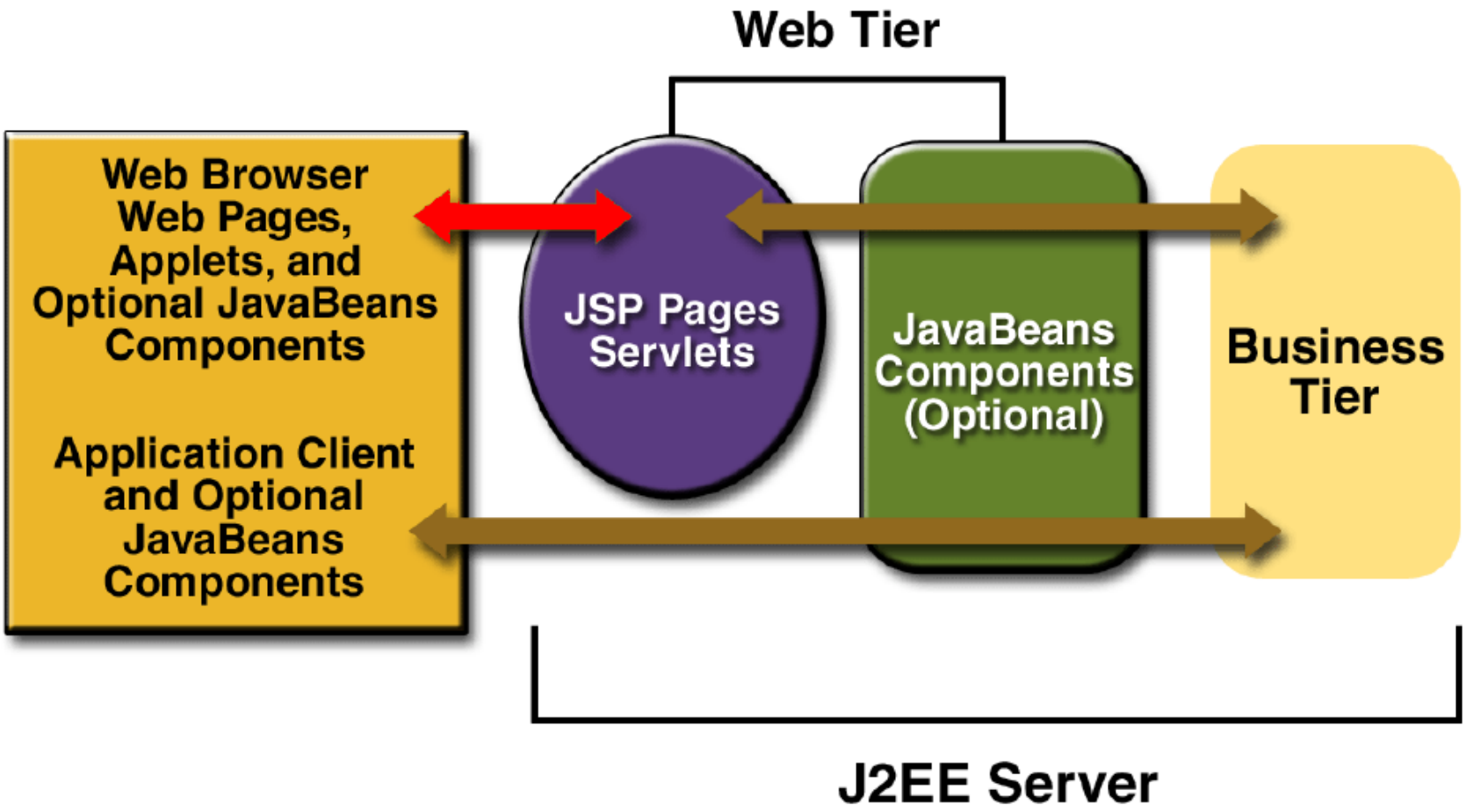
Client tier — Web tier — Business tier



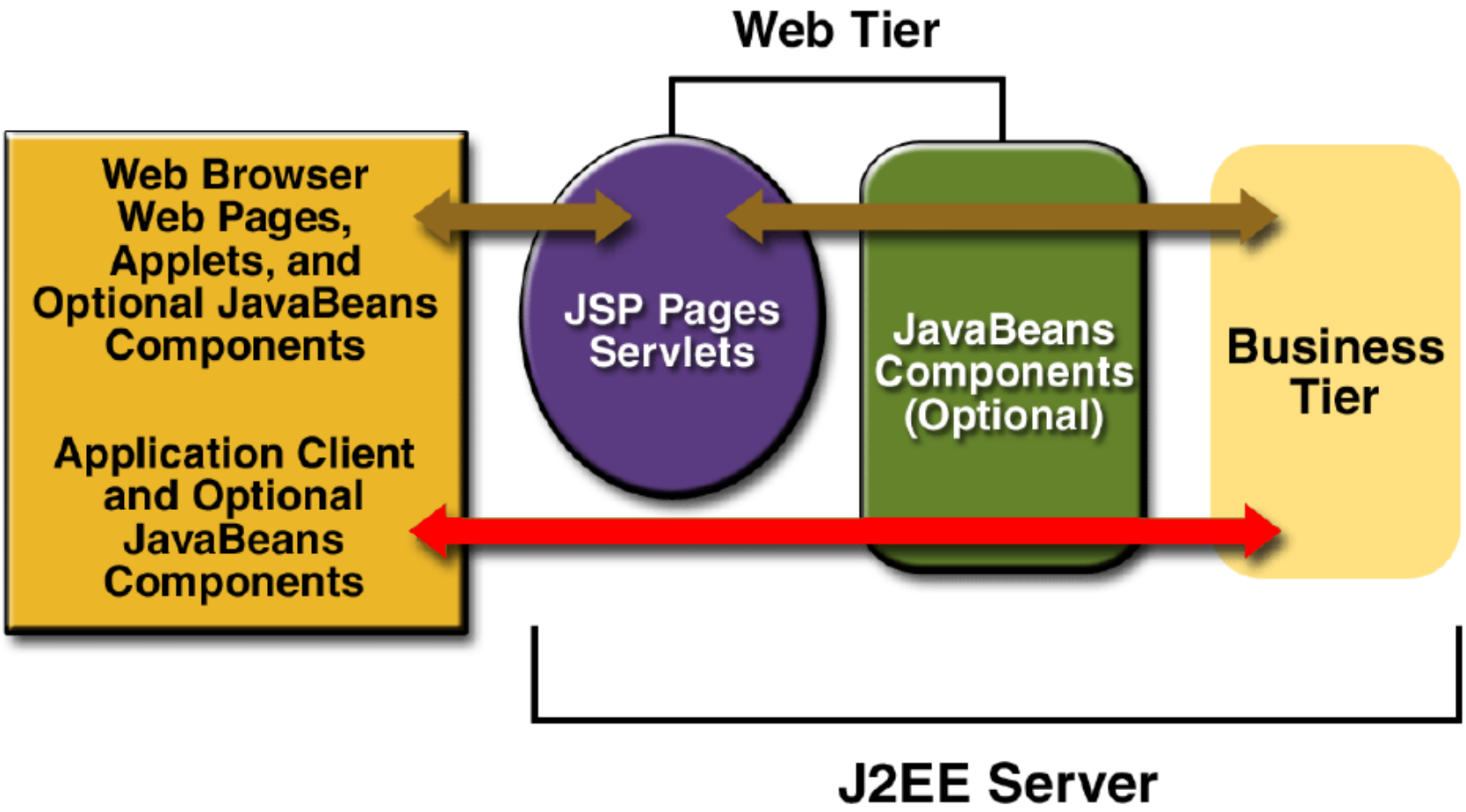
Client tier — Web tier — Business tier



Client tier — Web tier — Business tier



Client tier — Web tier — Business tier



Business Components

Business code, which is logic that solves or meets the needs of a particular business domain such as banking, retail, or finance, is handled by enterprise beans running in the business tier.

Business Components

Business code, which is logic that solves or meets the needs of a particular business domain such as banking, retail, or finance, is handled by enterprise beans running in the business tier.

An enterprise bean receives data from client programs, processes it (if necessary), and sends it to the enterprise information system tier for storage.

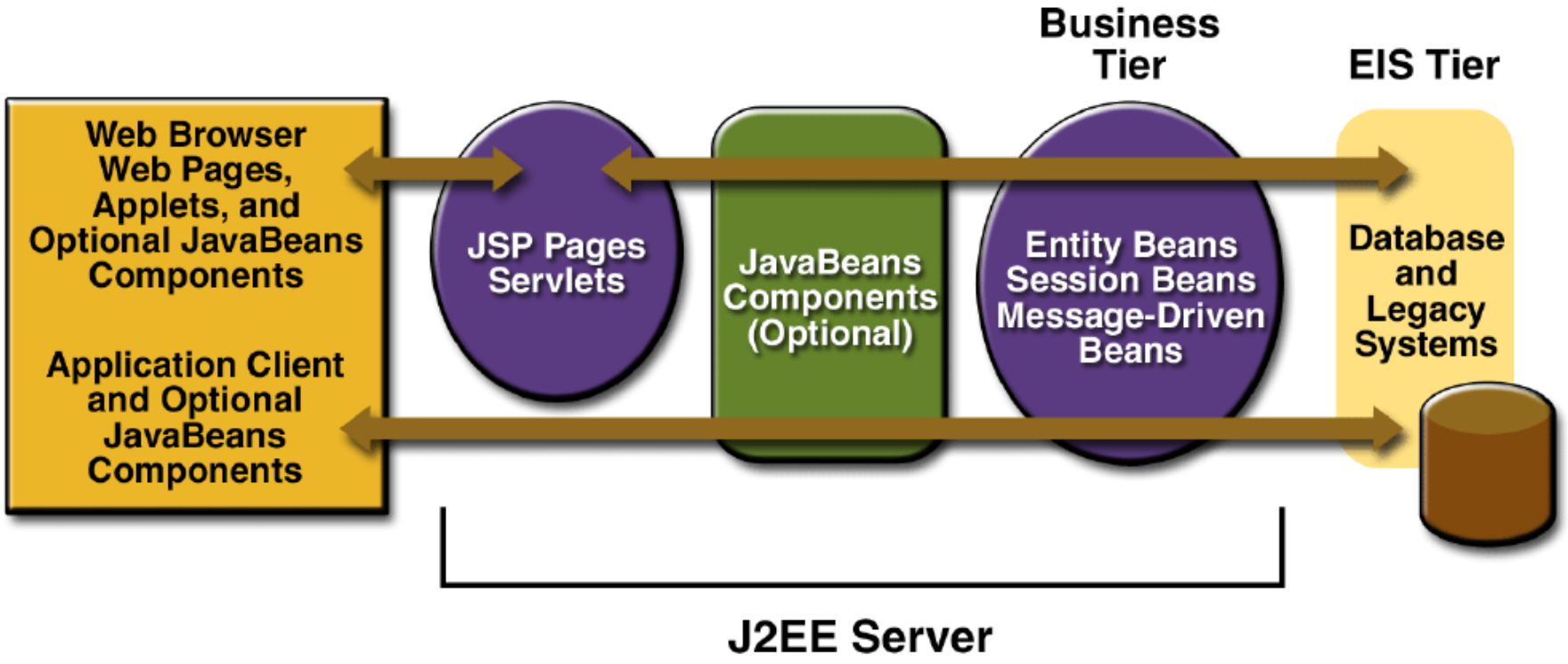
Business Components

Business code, which is logic that solves or meets the needs of a particular business domain such as banking, retail, or finance, is handled by enterprise beans running in the business tier.

An enterprise bean receives data from client programs, processes it (if necessary), and sends it to the enterprise information system tier for storage.

An enterprise bean also retrieves data from storage, processes it (if necessary), and sends it back to the client program.

Business and EIS tiers



Enterprise Java beans

There are three kinds of enterprise beans: session beans, entity beans, and message-driven beans.

Enterprise Java beans

There are three kinds of enterprise beans: session beans, entity beans, and message-driven beans.

A session bean represents a transient conversation with a client. When the client finishes, the session bean and its data are gone.

Enterprise Java beans

There are three kinds of enterprise beans: session beans, entity beans, and message-driven beans.

A session bean represents a transient conversation with a client. When the client finishes, the session bean and its data are gone.

In contrast, an entity bean represents persistent data stored in one row of a database table. If the client terminates or if the server shuts down, the underlying services ensure that the entity bean data is saved.

A message-driven bean combines features of a session bean and a Java Message Service (JMS) message listener, allowing a business component to receive JMS messages asynchronously.

Enterprise Information System Tier

The enterprise information system tier handles enterprise information system software and includes enterprise infrastructure systems such as enterprise resource planning (ERP), mainframe transaction processing, database systems, and other legacy information systems.

Enterprise Information System Tier

The enterprise information system tier handles enterprise information system software and includes enterprise infrastructure systems such as enterprise resource planning (ERP), mainframe transaction processing, database systems, and other legacy information systems.

J2EE application components might need access to enterprise information systems for database connectivity, for example.